SMART CONTRACTS

WHAT DID I LEARN ABOUT ETHEREUM SMART CONTRACTS?

Gesiel Rosa
@gesielrosa

ROADMAP

- Blockchain basics
- Ethereum Virtual Machine
- Smart Contract basics
- Solidity
- Compile and deploy
- Interaction (DApp)

BLOCKCHAIN

BLOCKCHAIN

- Transactions
- Blocks

EVM

ETHEREUM VIRTUAL MACHINE

ETHEREUM VIRTUAL MACHINE

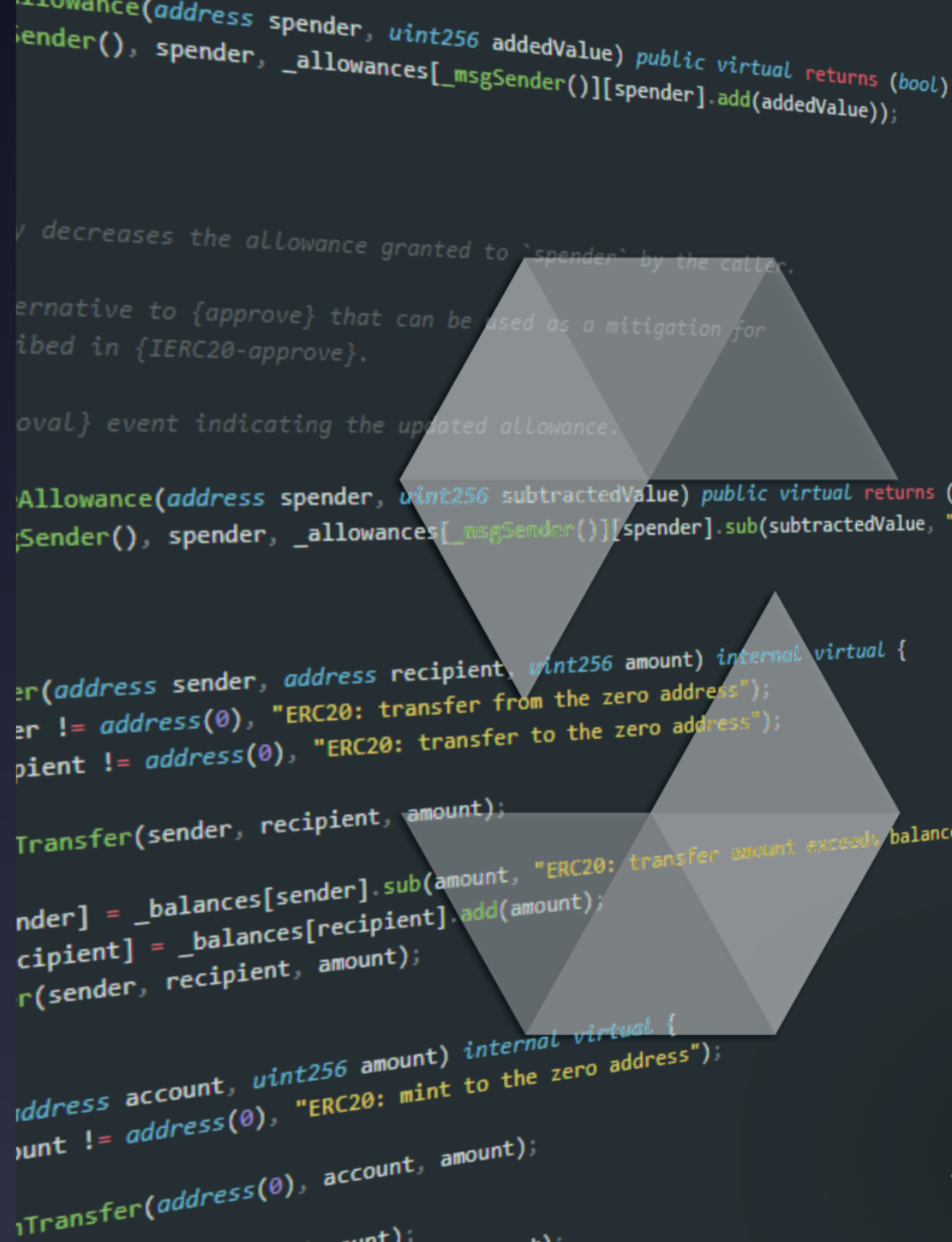
- Runtime environment for smart contracts
- Accounts
    - External - free
    - Contract - paid
- Transactions
- Gas

SMART CONTRACTS

SMART CONTRACTS

- Account with codes
- Enable on blockchain
- Transaction changes account data
- DApp

SOLIDITY



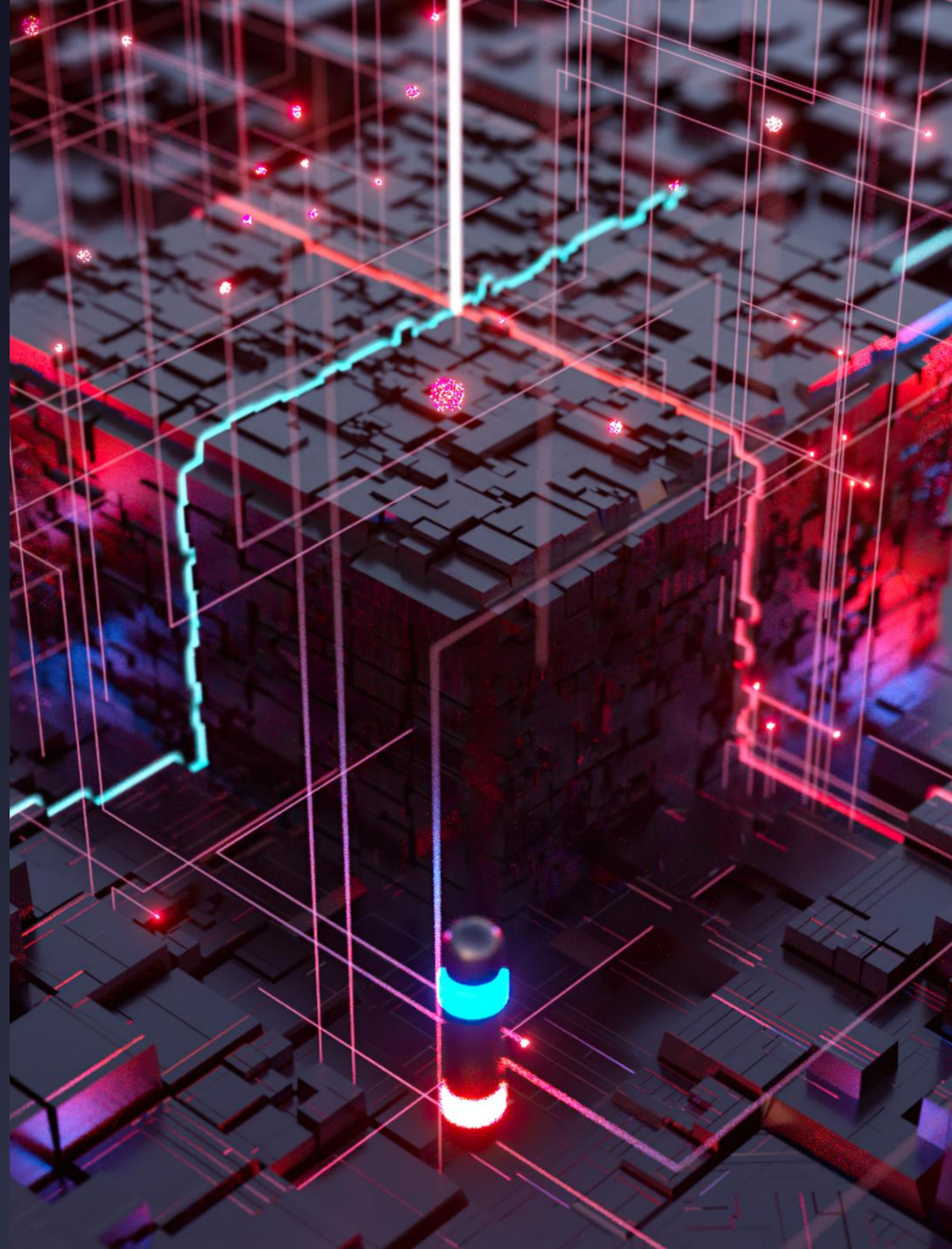
SOLIDITY

- Influenced by C ++, Python and JavaScript
- Technical characteristics
    - Statically typed
    - supports inheritance
    - supports libraries
    - supports complex user-defined type

CODE

```
// SPDX-License-Identifier: GPL-3.0

pragma solidity >=0.7.0 <0.9.0;

contract Storage {

    uint256 number;

    function store(uint256 num) public {
        number = num;
    }

    function retrieve() public view returns (uint256){
        return number;
    }
}
```

COMPILE AND DEPLOY



COMPILE

- Compiler: solc
    - NodeJS + solcJS
- Truffle Suite
- Input: contract.sol (solidity source code)
- Output: contract.json
    - Bytecode, ABI

DEPLOY

- contract.json
- Network (mainnet or testnet)
    - Ganache
- Account (external)
- Truffle Suite (again)
- Return: contract address

DAPP

DECENTRALIZED APPLICATIONS

DAPP

- Contract ABI and Address
- web3.js
    - Accounts, contracts
- MetaMask (provider/wallet)
- Contract instance
- Interact!!!

EXAMPLE DAPP

HTTPS://SC.RHIZOM.DEV/

- Interact with any Ethereum contract
- Angular + ~~@truffle/contract~~ + web3.js
- MetaMaskService
- ContractService
- SmartContract class

INTERACTING

```
const contract = new SmartContract<IContract>(injector, ABI, address);

contract.init();

contract.instance.MyFunction()
        .then(result => {
          console.log(result);
        })
        .catch(err => {
          console.log(err);
        });
```

THANKS